www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Applying clustering and folding techniques to study performance issues on the NEMO global ocean model

HPC Knowledge Meeting'15

Miguel Castrillo, Oriol Tintó, Harald Servat,
George S. Markomanolis, Kim Serradell

Barcelona, 3 February 2015

NEMO Model

- **N**ucleus for **E**uropean **M**odelling of the **O**cean

  – State-of-the-art modelling framework for oceanographic research, operational oceanography seasonal forecast and climate studies.

  – Created for studying the ocean and its interactions with the others components of the earth climate system over a wide range of space and time scales.

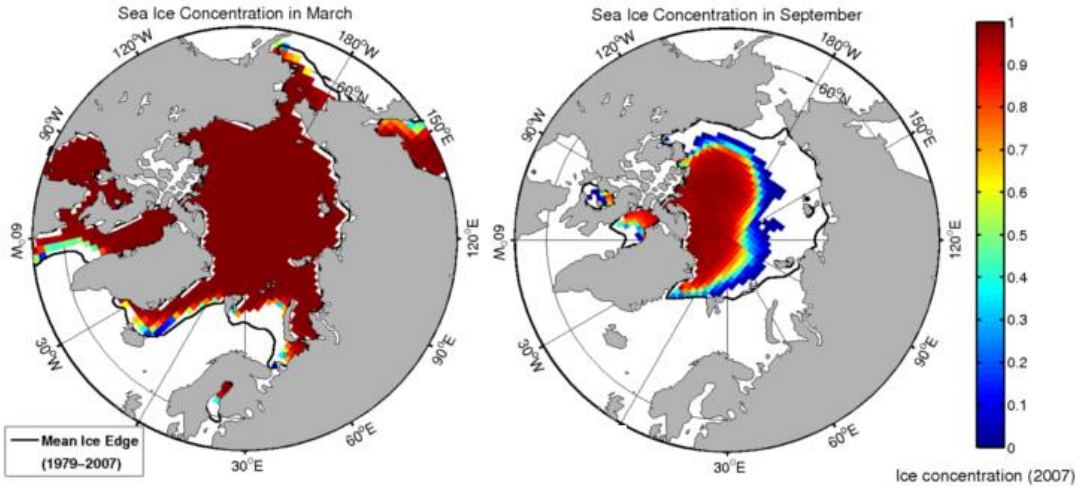
- Controlled by a European Consortium, formed by CNRS, NERC, UKMO, Mercator-OCEAN, CMCC, INGV

NEMO Model

- Composed of "engines" nested in an "environment".
  - The "engines" provide numerical solutions of ocean, sea-ice, tracers and biochemistry equations and their related physics.
    - Blue Ocean
    - White Ocean
    - Green Ocean
  - The "environment" consists of the pre- and post-processing tools, the interface to the other components of the Earth System, the user interface, computer dependent functions, documentation.
    - Adaptative mesh refinement (AGRIF)
    - Assimilation component

NEMO Model

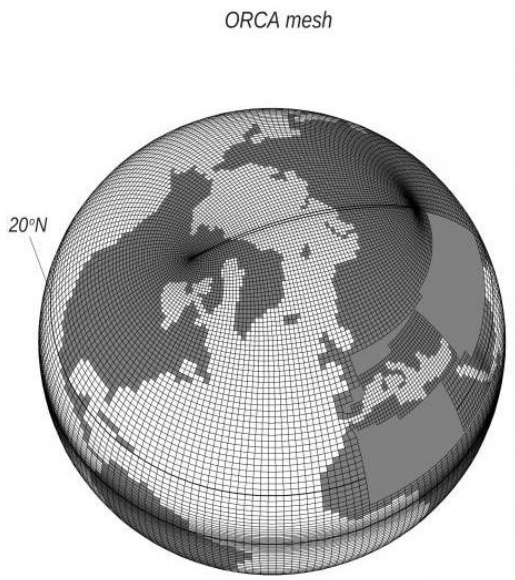
- Within NEMO, the ocean can be interfaced with a sea-ice model (LIM v2 and v3), passive tracer and biogeochemical models (TOP).

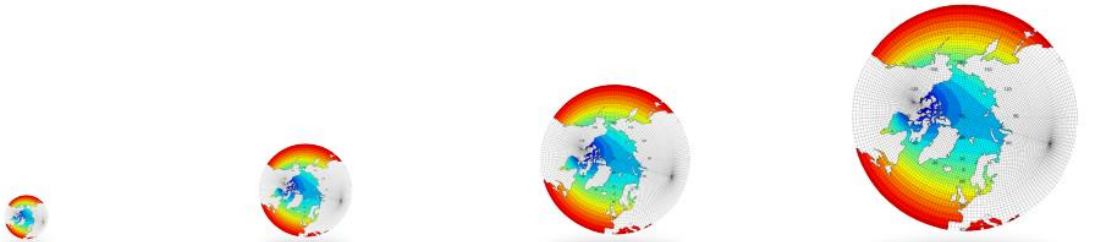


- Via the OASIS coupler, it can be interfaced with several atmospheric general circulation models.
- It also support two-way grid embedding via the AGRIF software.

NEMO Configurations

- ORCA (tripolar grid with climatological forcing)
  - ORCA2L31: 182 x 149 x 31 = 0,84 M
  - ORCA1L46: 362 x 292 x 46 = 4,86 M
  - ORCA05L46 : 722 x 511 x 46 = 16,97 M
  - ORCA025L46 : 1442 x 1021 x 46 = 67,72 M
  - ORCA1/16L100: 5762 x 3133 x 100 = 1.805,23 M





ORCA 2
550 MB of memory
8 CPU hours
10 Gigabytes of output (daily)

ORCA ¼
47 Gigabytes of memory
3500 CPU hours
120 Gigabytes of output (daily)

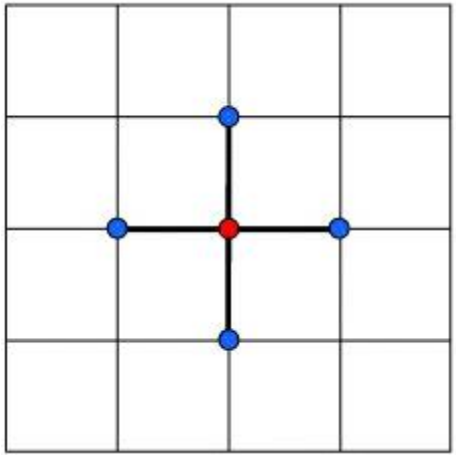
ORCA 1/12
414 Gigabytes of memory
90 000 CPU hours
1 Terabyte of output (daily)

ORCA 1/36
> 1 Terabytes of memory
~4 000 000 CPU hours
> 5 Terabytes of output (daily)

- GYRE (idealized double gyres on a beta-plane with analytical forcing)
  - Customizable size

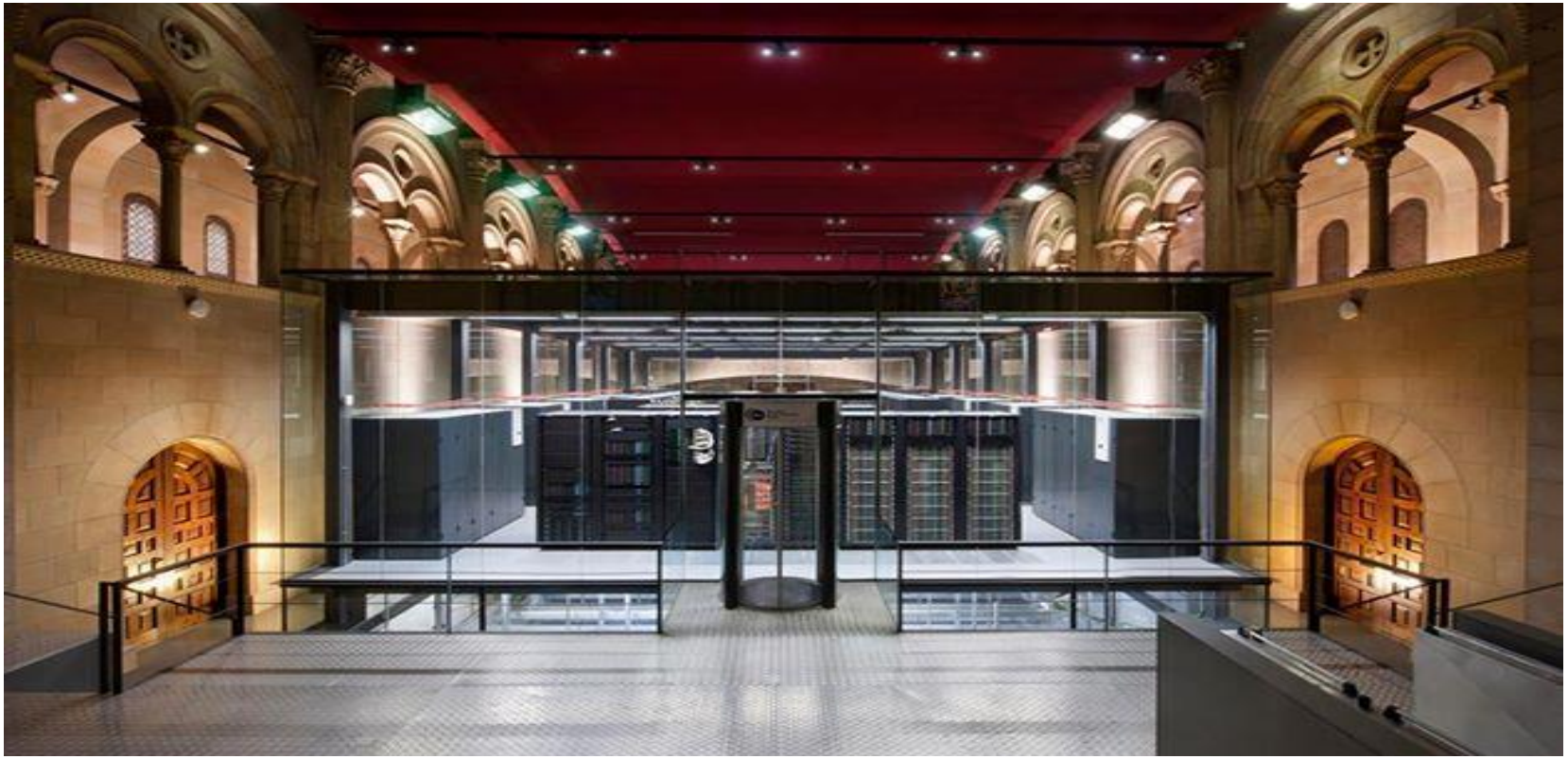
NEMO – Technical aspects

- Almost 170.000 lines of FORTRAN 90 code
- Parallelization based on domain decomposition through MPI.
- Mostly small stencil element calculations



- Our configuration
  - NEMO v3.6 trunk version
  - ORCA025 tripolar grid (¼ degree resolution)
  - LIM3 sea-ice model

MareNostrum III

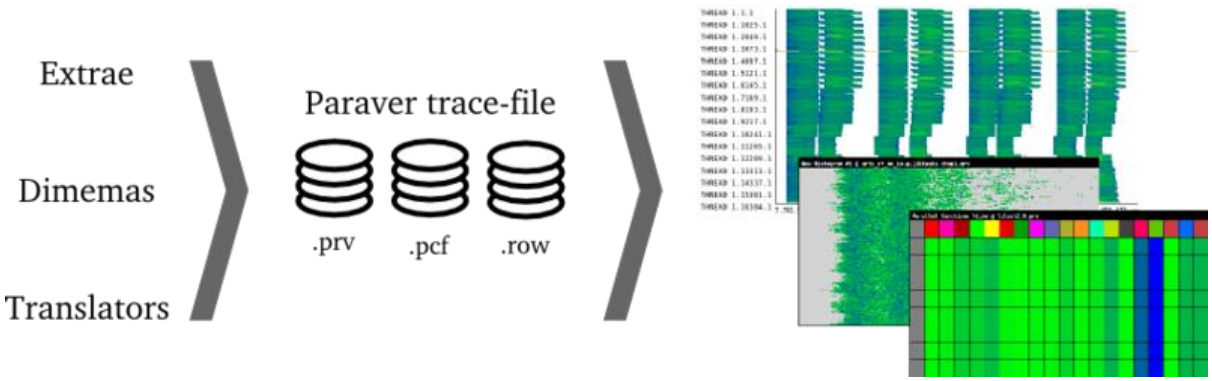


- 3,056 compute nodes
- 2x Intel SandyBridge-EP E5-2670 2.6 GHz
- 32 GB memory per node

- Infiniband FDR10
- OpenMPI 1.8.1
- ifort 13.0.1

BSC Computer Sciences department – Performance Tools

- Since 1991
- Based on traces
- Open Source: http//:www.bsc.es/paraver
- Extrae: package that generates Paraver trace-files for a post-mortem analysis.
- Paraver: trace visualization and analysis browser.
  - Includes trace manipulation: Filter, cut traces
- Dimemas: message passing simulator



Extrae

- BSC instrumentation package

- When/Where
  - Parallel programming model runtime
  - Selected user functions
  - Periodic samples
  - User events

- Additional information
  - Counters

Paraver

- Every behavioral aspect/metric of a thread can be described as a function of time
- Those functions of time can be rendered into a 2D image
- Statistics can be computed for each possible value or range of values of that function of time
  - Tables: Profiles and histograms
- Types of functions
  - Categorical
    - State, user function…
  - Logical
    - In specific user function, in MPI call…
  - Numerical
    - IPC, L2 miss ratio, duration of computation burst or MPI call…

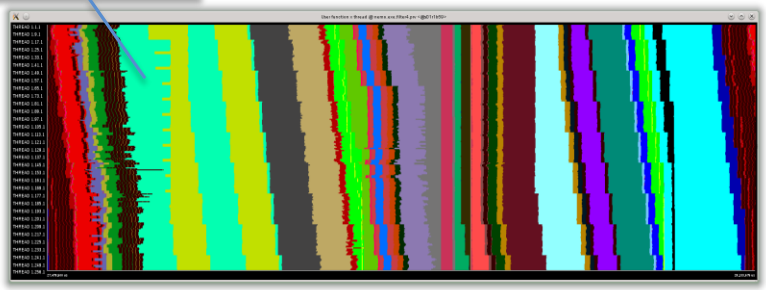
PARAVER trace analysis

Serial efficiency

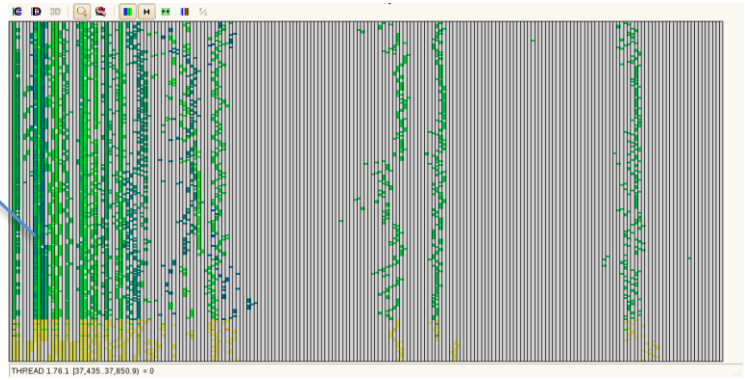
Each color represents a function

Functions instrumentation

Cores (parallel processes)

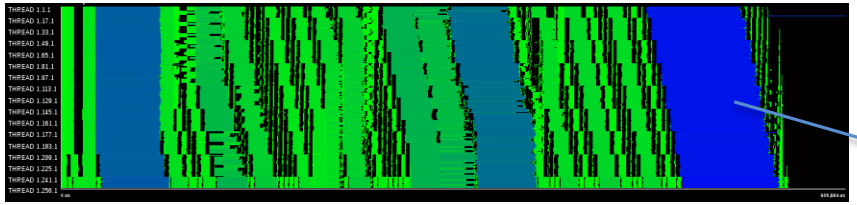


Correlation between useful duration and IPC



Darker color = Higher IPC

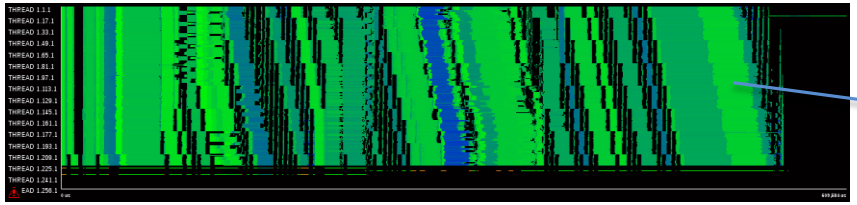
Useful duration - Time of computation bursts (computation between MPI calls)



Darker color = Longer duration

Columns = Ranges of computation burst duration
Vertical lines = Similar performance along the processes

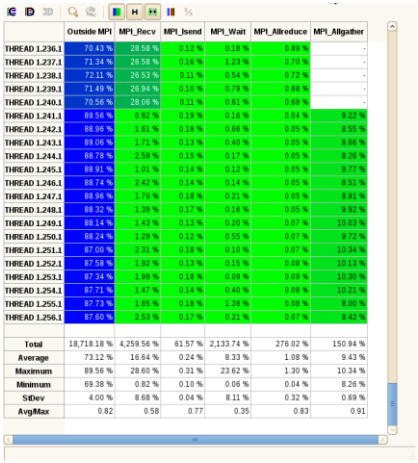
Useful IPC - Instructions per cycle



Lighter color = Less IPC

Horizontal axis -> Time component

Parallel efficiency



	Outside MPI	MPI_Recv	MPI_Isend	MPI_Wait	MPI_Allreduce	MPI_Allgather
THREAD 1.236.1	70.43 %	28.58 %	0.12 %	0.18 %	0.69 %	-
THREAD 1.237.1	71.34 %	26.58 %	0.16 %	1.23 %	0.70 %	-
THREAD 1.238.1	72.11 %	26.53 %	0.11 %	0.54 %	0.72 %	-
THREAD 1.239.1	71.49 %	26.94 %	0.10 %	0.79 %	0.68 %	-
THREAD 1.240.1	70.56 %	28.06 %	0.11 %	0.61 %	0.68 %	-
THREAD 1.241.1	89.56 %	0.82 %	0.19 %	0.16 %	0.04 %	9.22 %
THREAD 1.242.1	88.96 %	1.61 %	0.18 %	0.66 %	0.05 %	8.55 %
THREAD 1.243.1	89.06 %	1.71 %	0.13 %	0.40 %	0.05 %	8.66 %
THREAD 1.244.1	88.78 %	2.59 %	0.15 %	0.17 %	0.05 %	8.26 %
THREAD 1.245.1	88.91 %	1.01 %	0.14 %	0.12 %	0.05 %	9.77 %
THREAD 1.246.1	88.74 %	2.42 %	0.14 %	0.14 %	0.05 %	8.51 %
THREAD 1.247.1	88.96 %	1.70 %	0.18 %	0.21 %	0.05 %	8.91 %
THREAD 1.248.1	88.32 %	1.39 %	0.17 %	0.16 %	0.05 %	9.92 %
THREAD 1.249.1	88.14 %	1.43 %	0.13 %	0.20 %	0.07 %	10.03 %
THREAD 1.250.1	88.24 %	1.29 %	0.12 %	0.55 %	0.07 %	9.72 %
THREAD 1.251.1	87.00 %	2.31 %	0.18 %	0.10 %	0.07 %	10.34 %
THREAD 1.252.1	87.58 %	1.92 %	0.13 %	0.15 %	0.08 %	10.13 %
THREAD 1.253.1	87.34 %	1.99 %	0.18 %	0.09 %	0.09 %	10.30 %
THREAD 1.254.1	87.71 %	1.47 %	0.14 %	0.40 %	0.08 %	10.21 %
THREAD 1.255.1	87.73 %	1.65 %	0.18 %	1.36 %	0.08 %	8.00 %
THREAD 1.256.1	87.60 %	2.53 %	0.17 %	0.21 %	0.07 %	9.42 %
Total	18,718.18 %	4,259.56 %	61.57 %	2,133.74 %	276.02 %	150.94 %
Average	73.12 %	16.64 %	0.24 %	8.33 %	1.08 %	9.43 %
Maximum	89.56 %	28.60 %	0.31 %	23.62 %	1.30 %	10.34 %
Minimum	69.38 %	0.82 %	0.10 %	0.06 %	0.04 %	8.26 %
StDev	4.00 %	8.68 %	0.04 %	8.11 %	0.32 %	0.69 %
Avg/Max	0.82	0.58	0.77	0.35	0.83	0.91

MPI Stats reflect the percentage of time invested in computation for each thread.

Total stats give the communication efficiency and the load balance

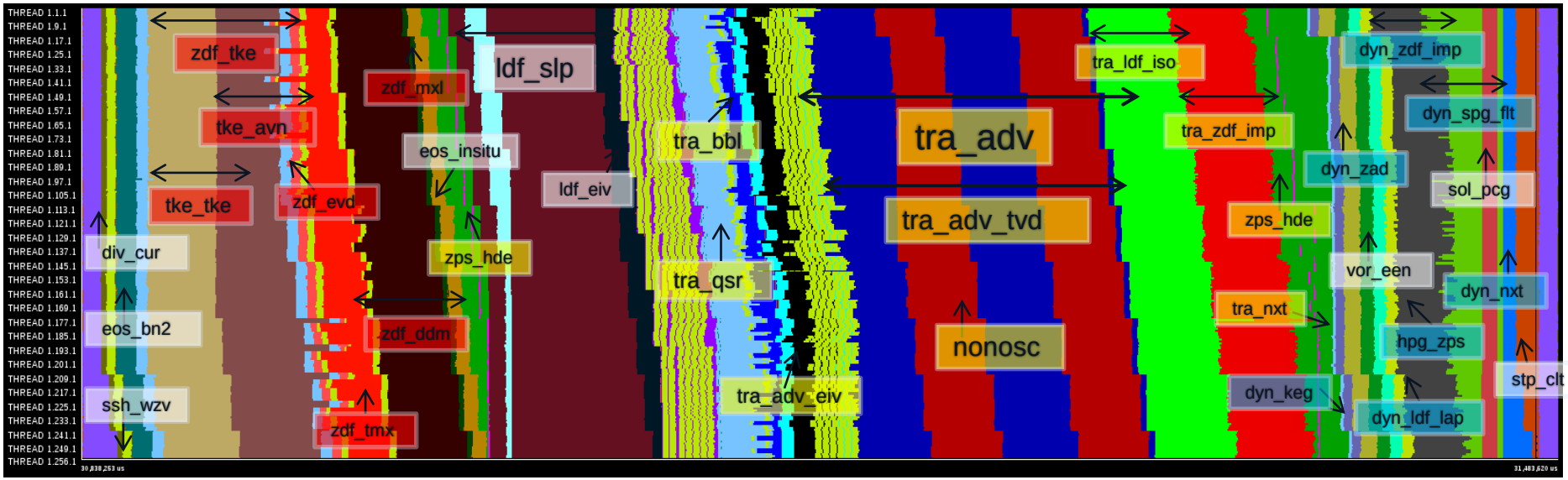
Trace functions instrumentation

- **tra:** Ocean tracers
  - **tra_qsr**: Solar radiation penetration → **trc_oce_rgb**: Visible
  - **tra_bbl:** Bottom boundary layer.
  - **tra_adv:** Advection .
    → **trc_adv_eiv**: Eddy Induced Velocity.
    → **trc_adv_tvd**: Eddy Induced Velocity → **nonosc**
  - **tra_ldf_iso:** Lateral Diffusion
  - **tra_zdf_imp:** Vertical diffusion
  - **zps_hde:** Horizontal Derivative in zps-coordinate
  - **tra_nxt:** Tracer time evolution

- **vor_een:** Vorticity Term with EEN scheme
- **sol_pcg**: Preconditioned Conjugate Gradient (Elliptic Solver)
- **ssh_wzv**: Sea surface height evolution and vertical velocity → Horizontal divergence and relative velocity (**div_cur**)
- **ldf**: Lateral ocean physics
  - **ldf_slp:** Direction slopes
  - **ldf_eiv:** Eddy induced velocity

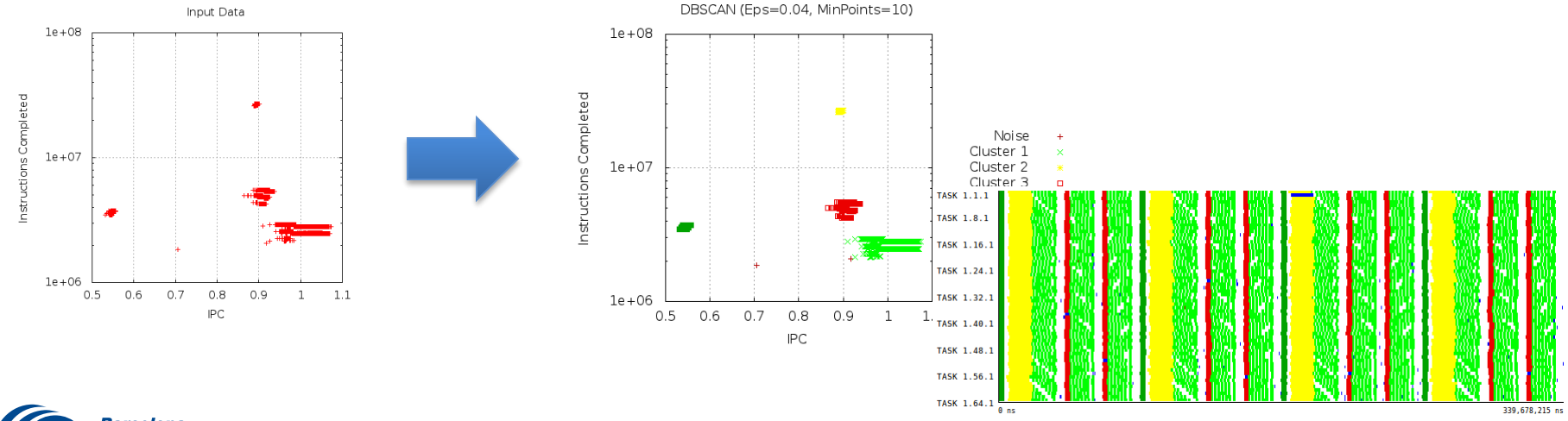
- **dyn:** Ocean dynamics
  - **dyn_adr:**
    → **dyn_keg**: Kinetic Energy Gradient term.
    → **dyn_zad**: Vertical advection term
  - **hpg_zps:** Z-coordinate with partial step
  - **dyn_zdf_imp:** Vertical diffusion term
  - **dyn_ldf_lap:** Lateral diffusion term – Iso-level laplacian oper.
  - **dyn_spg_flt:** Surface pressure gradient – Filtered free surfc.

- **zdf:** Vertical Ocean Physics
  - **zdf_tke**: Turbulent kinetic energy
    → **tke_tke**:
    → **tke_avn**:
  - **zdf_evd:** Enhanced vertical diffusion
  - **zdf_tmx:** Tidal mixing
  - **zdf_ddm:** Double diffusion mixing.
  - **zdf_mxl:** Mixing.



Clustering (Cluster analysis)

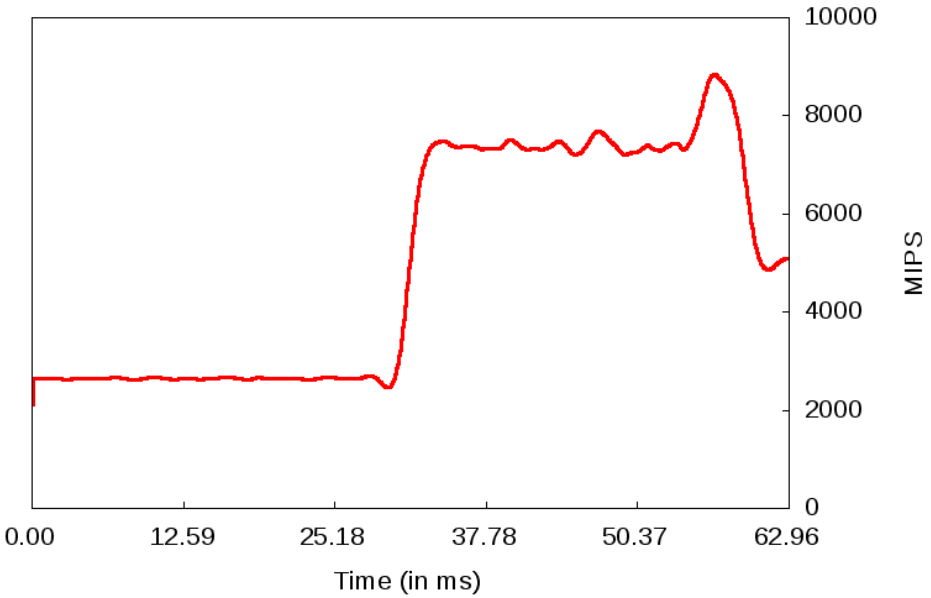
- Data mining technique used for the classification of data
- Applied to detect different trends in the computation regions (CPU bursts) of a given application.
- Consists in identifying computation regions of similar behavior.
- Similarity intended in terms of duration or hardware counter reduced metrics.
  - For example, Instructions Completed and Instructions Per Cycle (IPC)
- DBSCAN algorithm (Density-Based Spatial Clustering of Applications with Noise)
  - Two parameters: Epsilon (distance) and Minimum points (density)



Motivation

Which is the performance of routine A?
5000 MIPS, 850 MFLIPS
(on an Intel SandyBridge @ 2.6GHz)

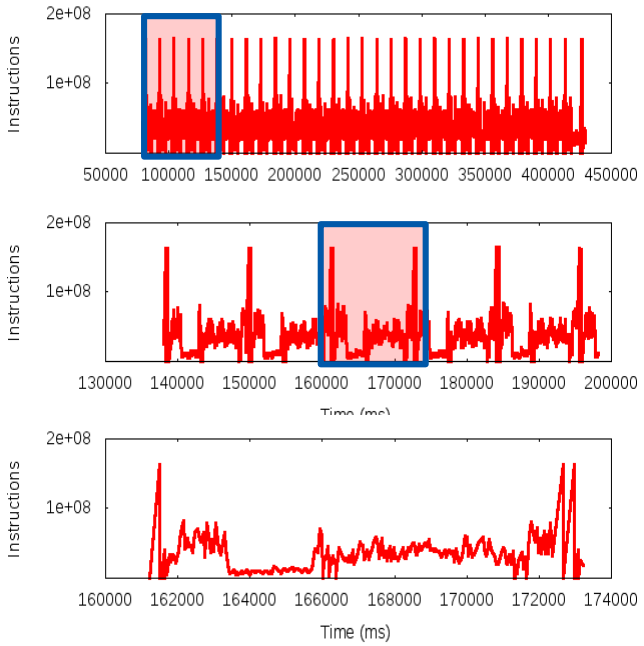
How is the performance ***within*** routine A?



Application behavior

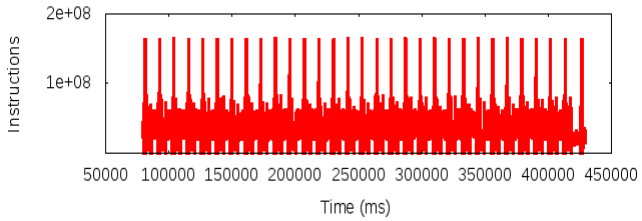
- What if the analyst knows ε about the application?
  - Instrument everything / selectively?
  - Use high frequency rates?

- HPC / Scientific applications
  - Repetitive nature

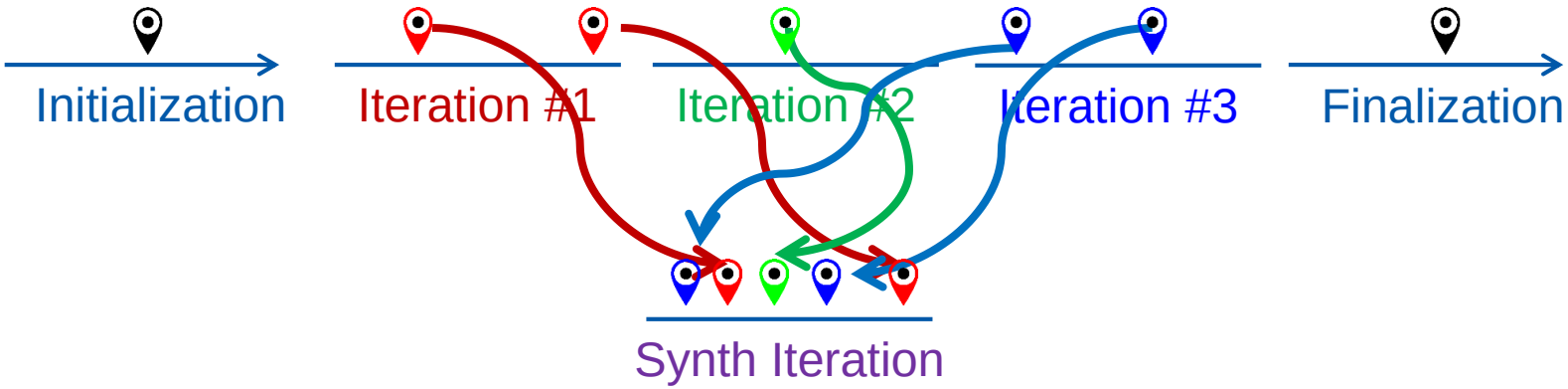


Folding

- HPC / Scientific applications
  - Repetitive nature



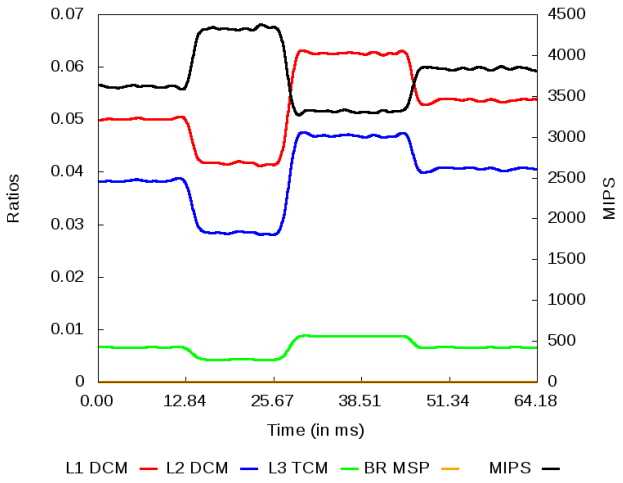
- Instantaneous metrics with minimum overhead
  - Combine instrumentation and sampling
    - Instrumentation delimits regions (routines, loops, …)
    - Sampling exposes progression within a region
  - Captures performance counters and call-stack references



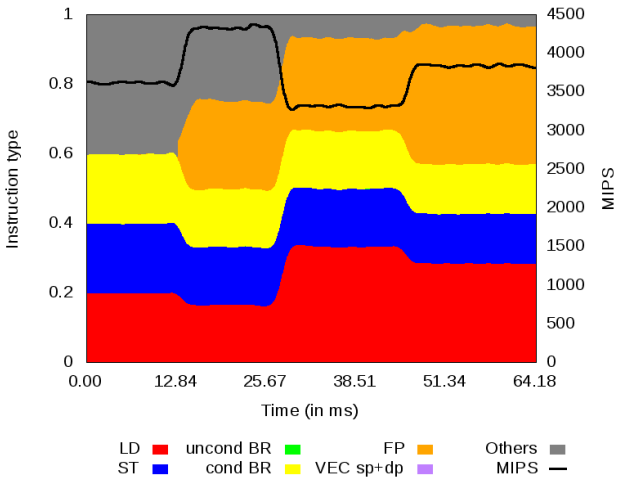
Combining performance counters

- Need to correlate multiple performance metrics
  - Apply models based on performance counters
    - Calculate ratio of counter rates

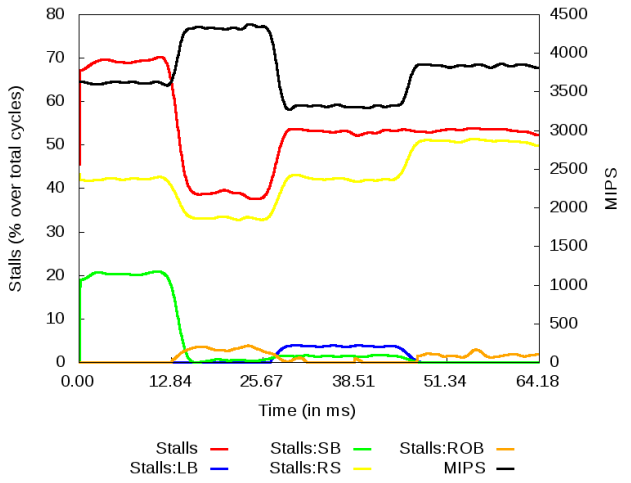
Architecture impact



Instruction mix

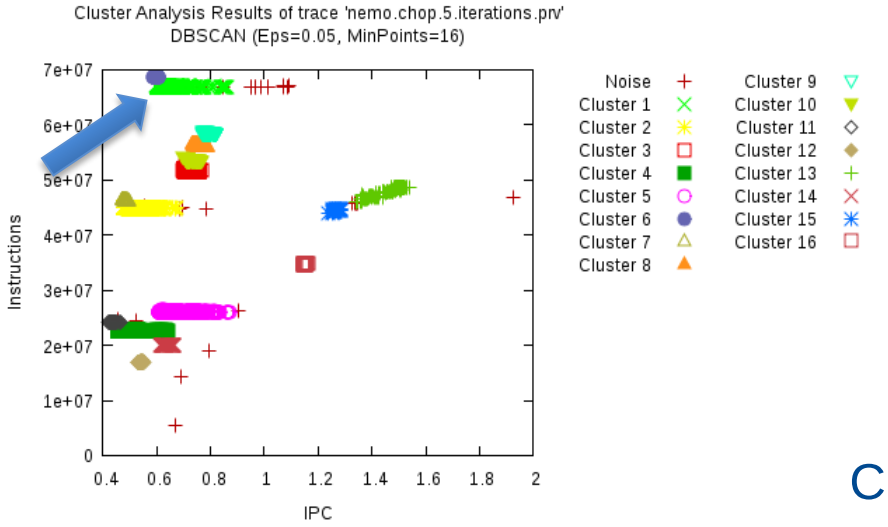


Stall distribution



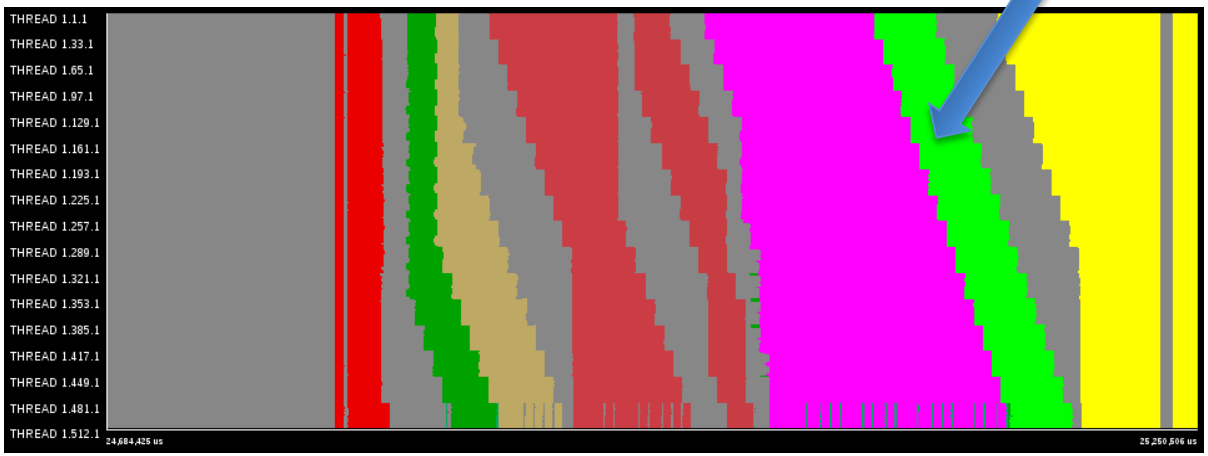
Results: Clustering on NEMO

Scatter Plot of Clustering Metrics



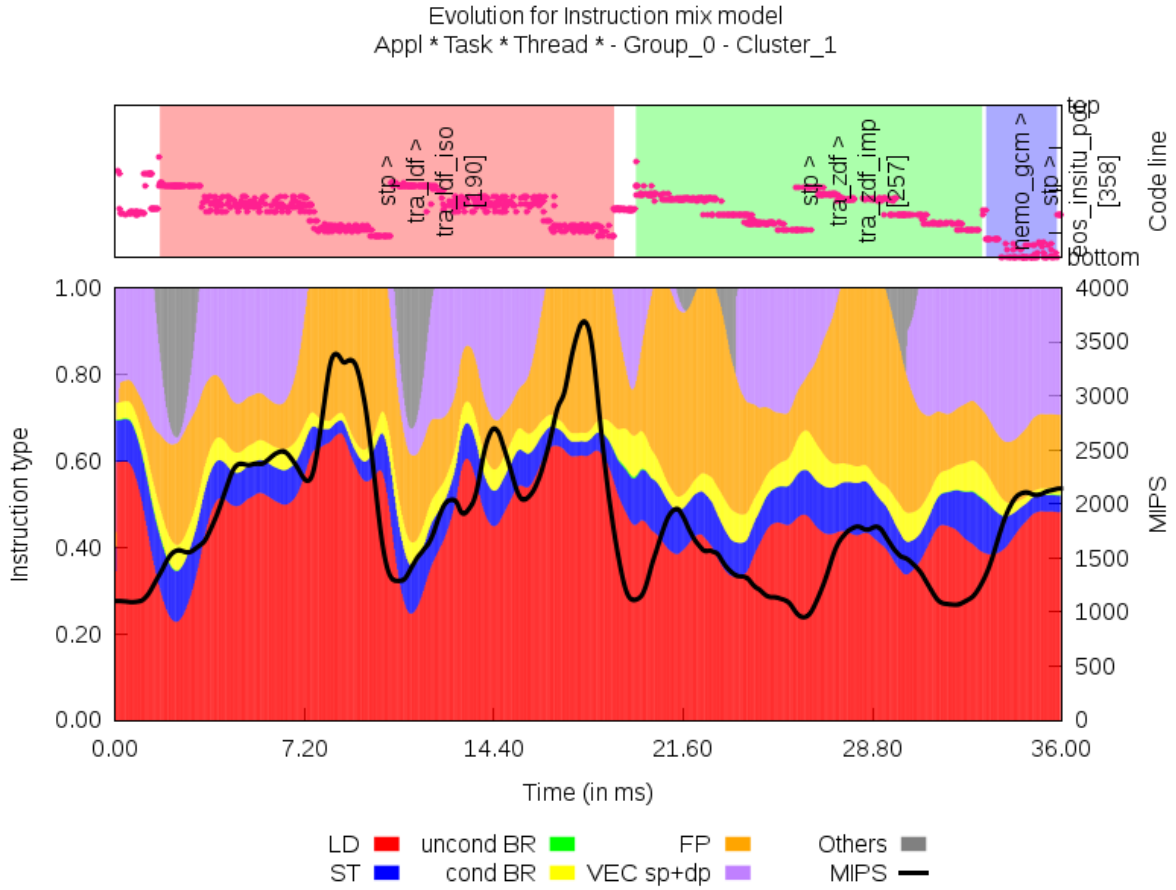
Cluster 1 has around 70 million instructions and a low IPC value

Clusters Distribution Over Time



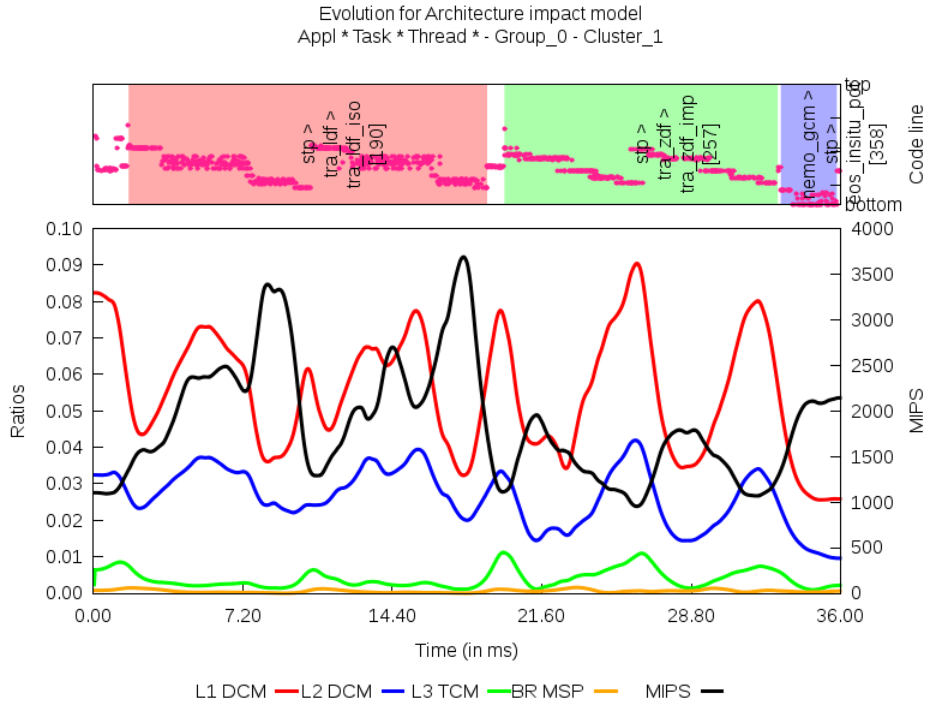
Results: Cluster 1

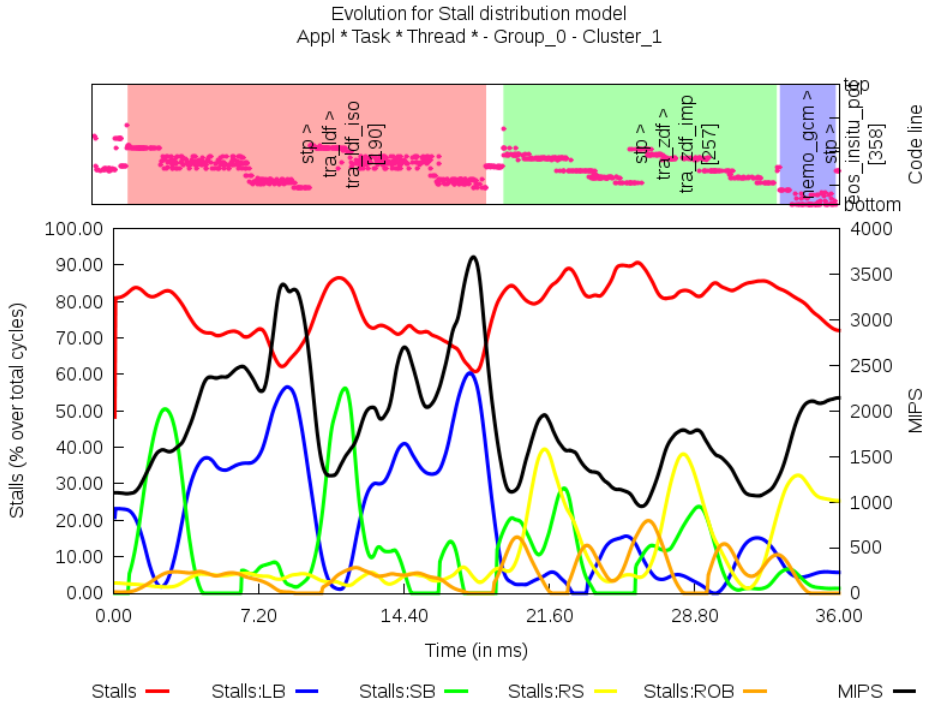
- Instruction mix distribution
- Repetitive patterns can be seen in the code line plots



Results: Cluster 1

- Views of architecture impact and stalls distribution
- In the Architecture impact view there is not always a correlation between data cache misses and MIPS
  - In the performance peaks it exists that correlation
  - Between the peaks the stalls distribution plot explains the results

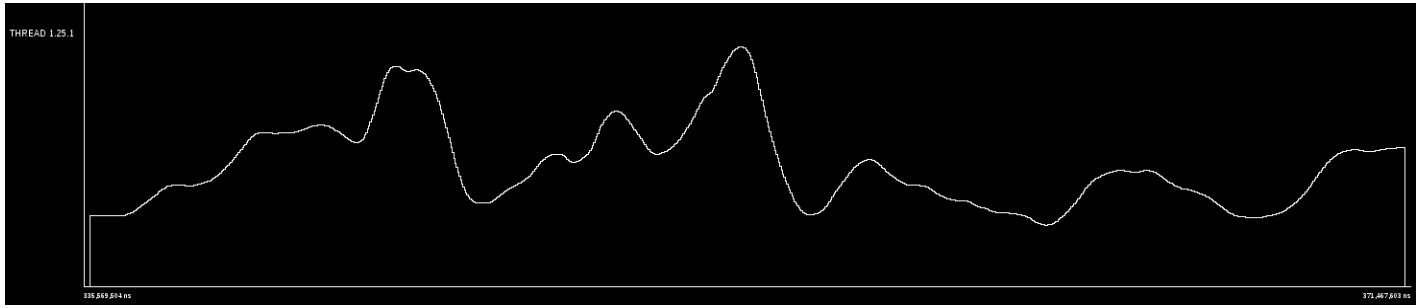




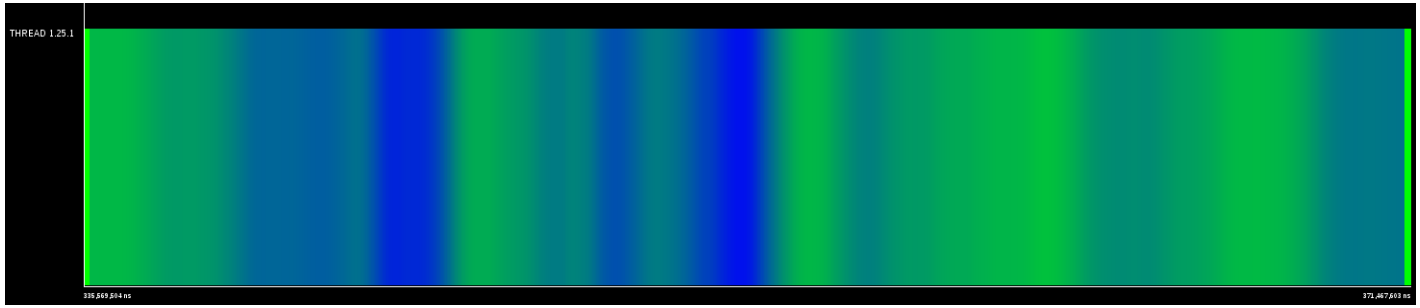
Results: Cluster 1

- It's easy to correlate different metrics with source code lines

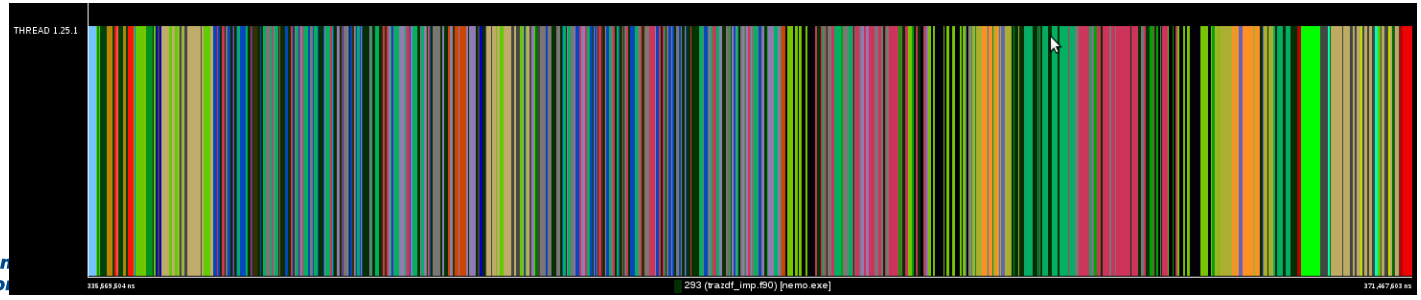
MIPS Function line



MIPS Gradient visualization

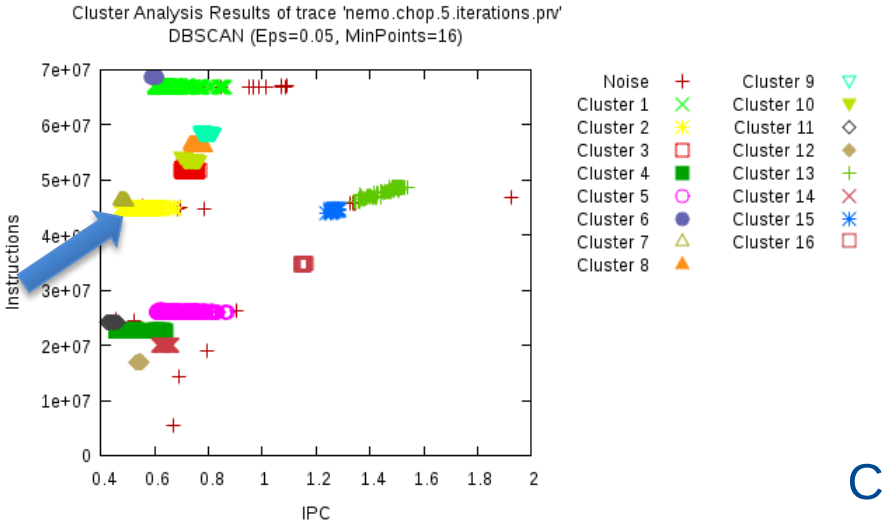


Routines and code lines



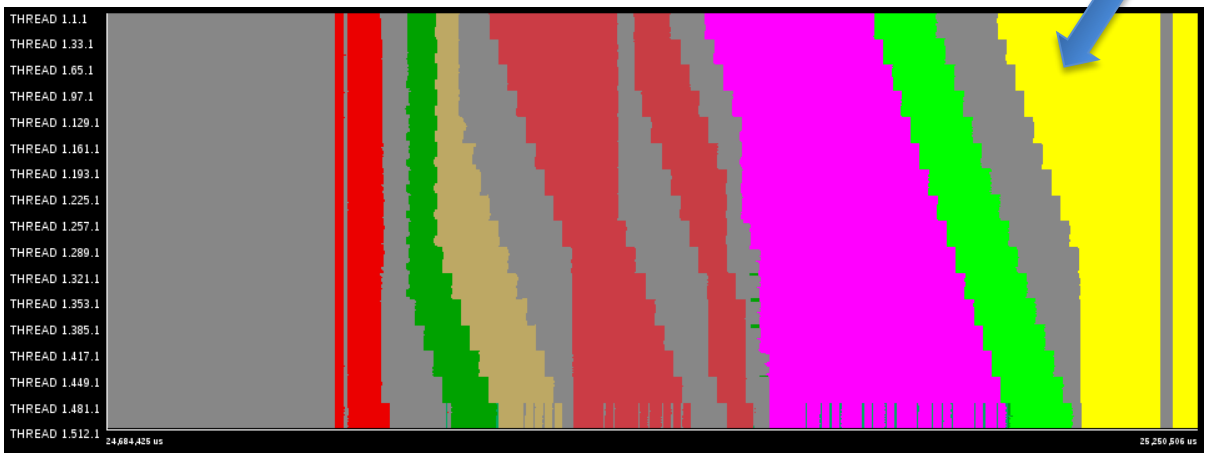
Results: Clustering on NEMO

Scatter Plot of Clustering Metrics



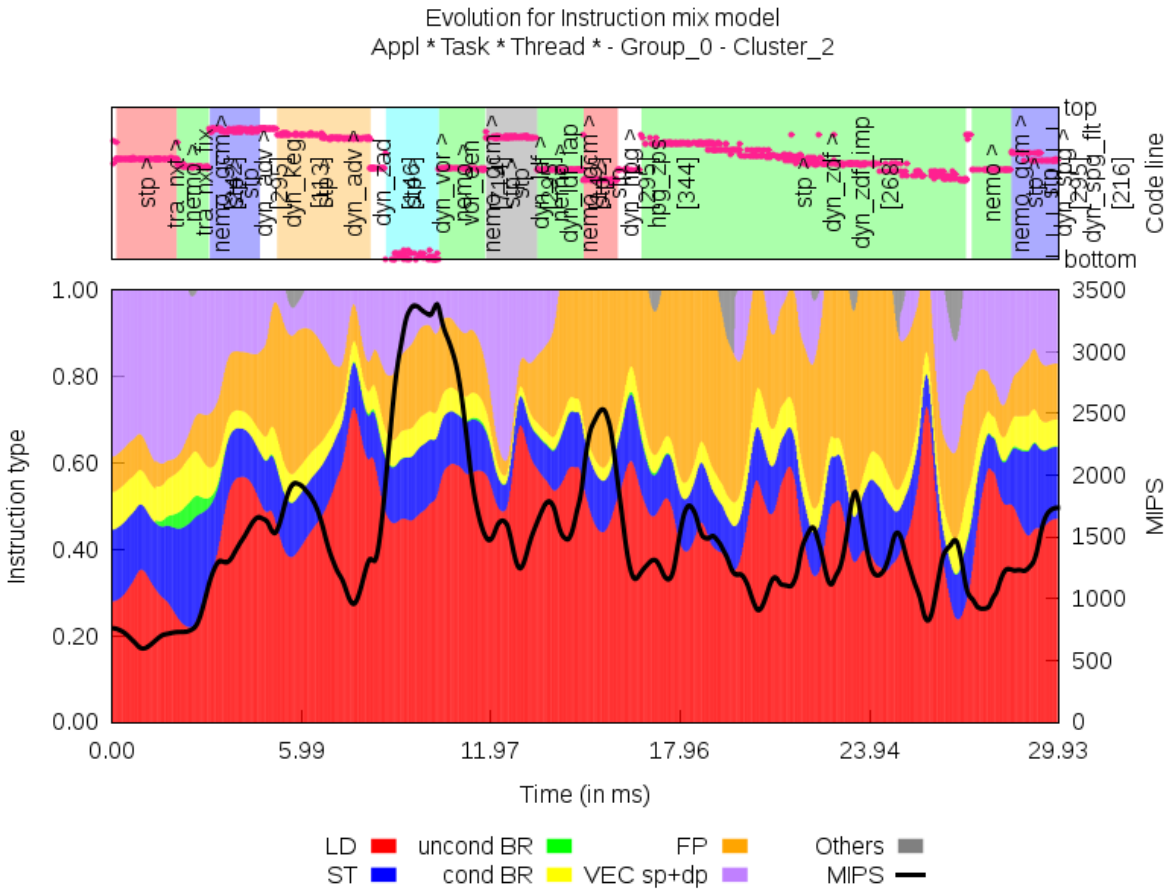
Cluster 2 has around 45 million instructions and a very low IPC value

Clusters Distribution Over Time



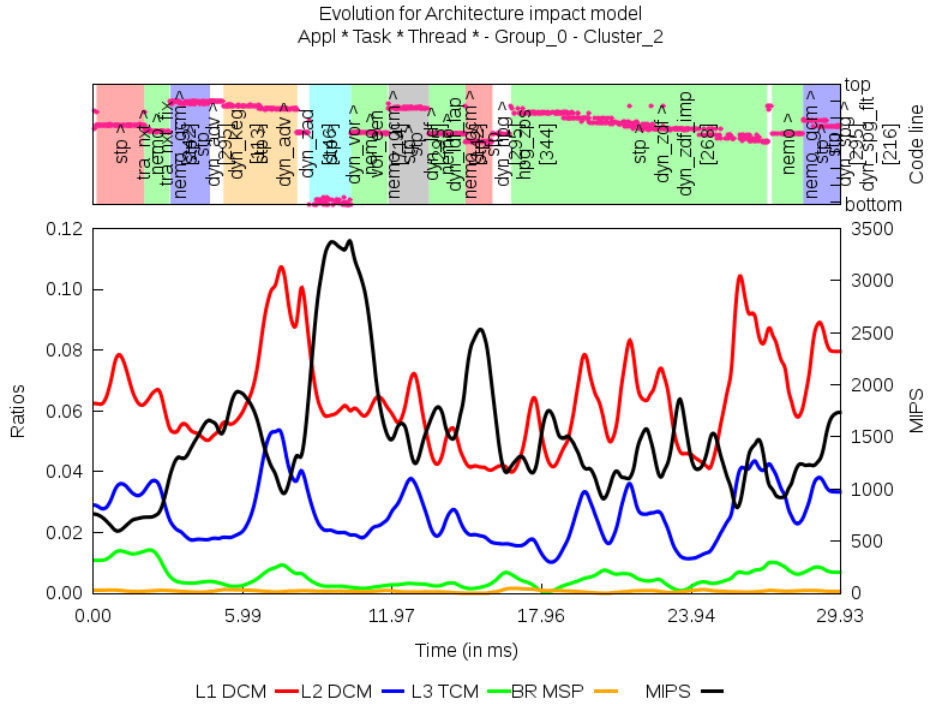
Results: Cluster 2

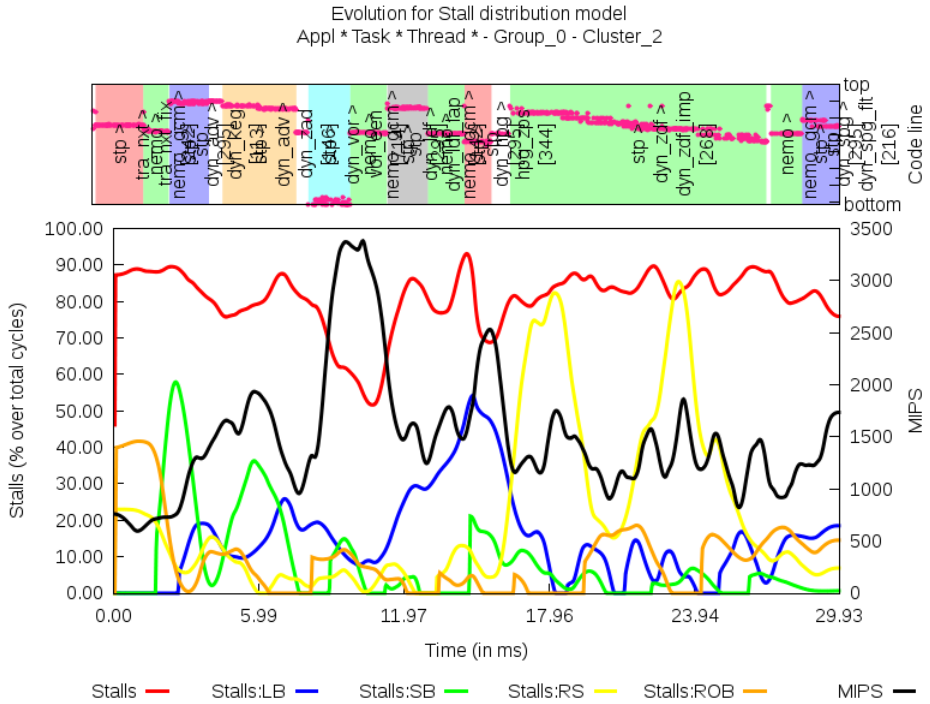
- In this cluster, a LD increase generally implies a decrease of the MIPS



Results: Cluster 2

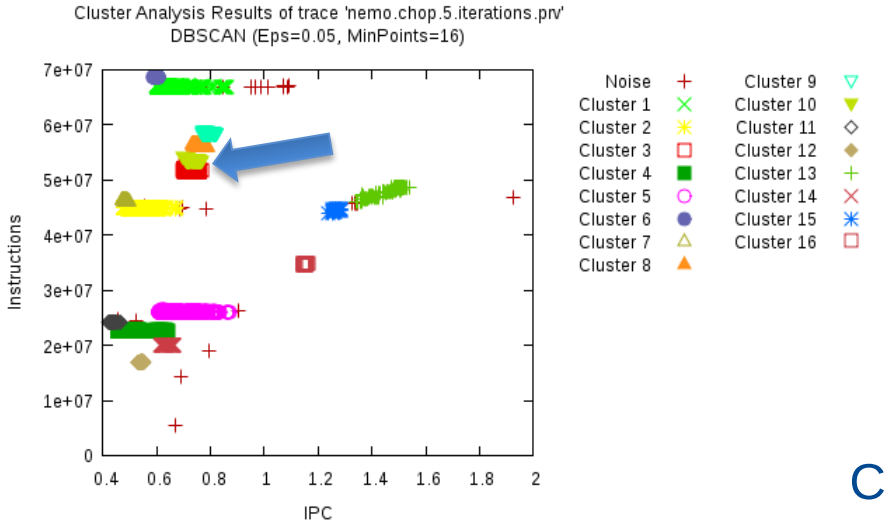
- The performance peak can be explained from the low cache misses ratio and stalls decrease
- The lower MIPS values come either from higher data cache misses ratios or more stalls percentage





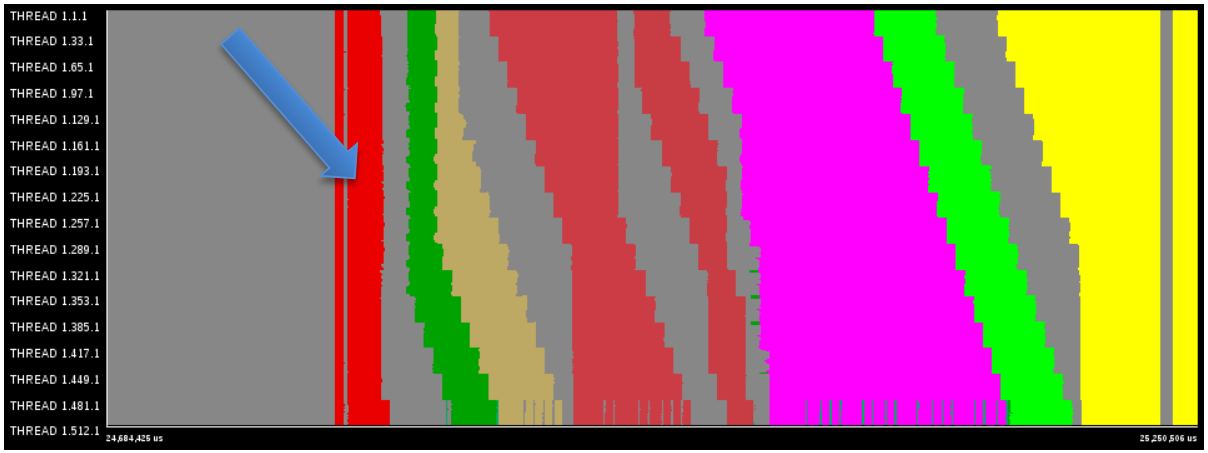
Results: Clustering on NEMO

Scatter Plot of Clustering Metrics



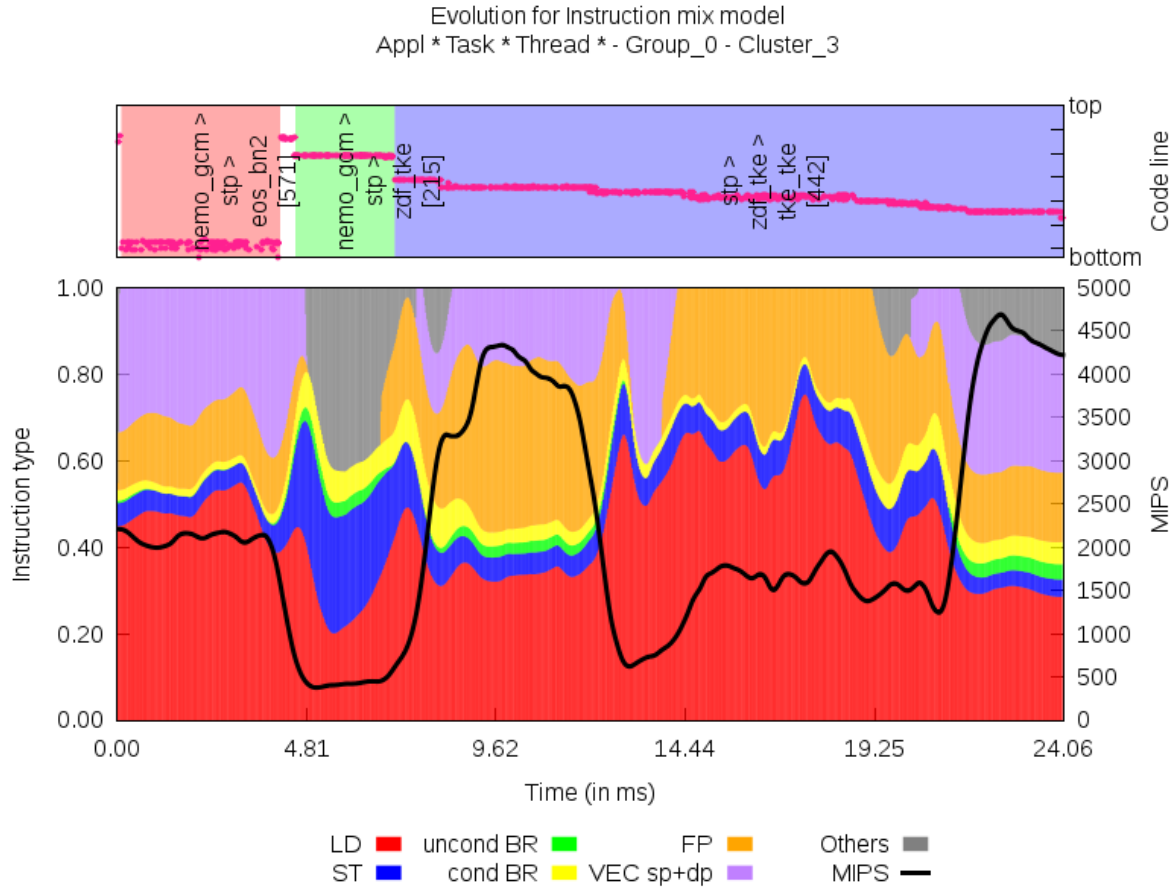
Cluster 3 has around 50 million instructions and a extremely low IPC value

Clusters Distribution Over Time



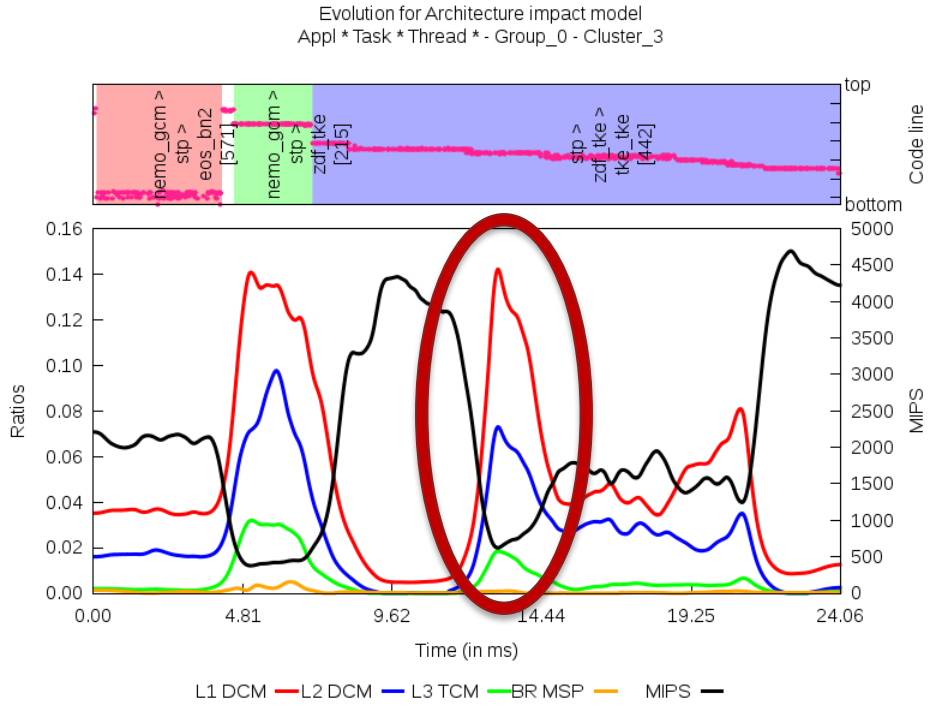
Results: Cluster 3

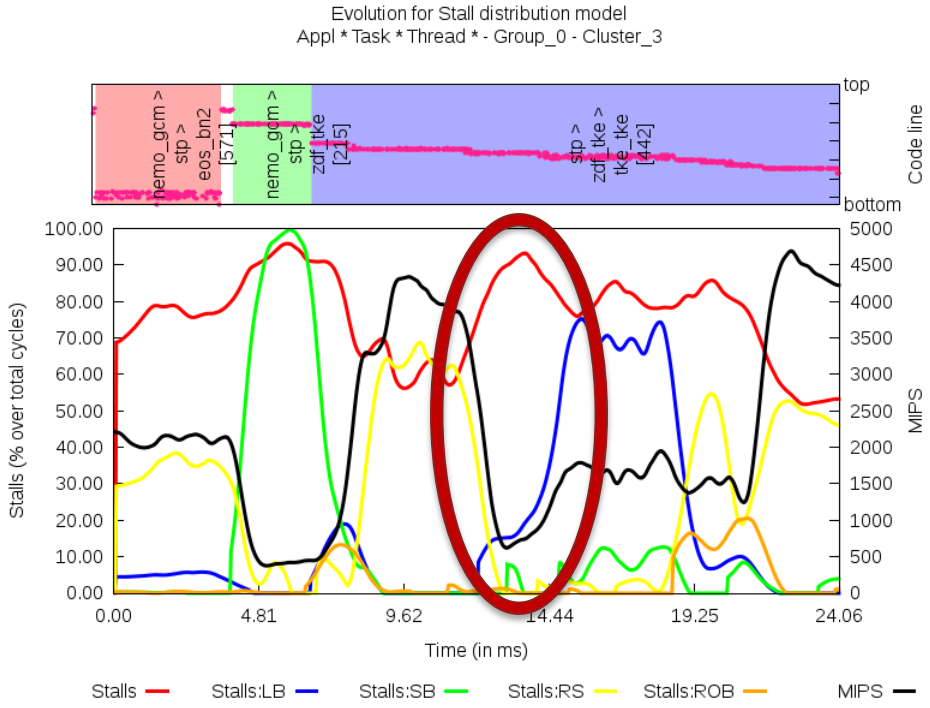
- The first performance decrease coincides with a lot of store instructions but also other not categorized instructions, and the second with an increase of load and vector operations



Results: Cluster 3

- The minimum value in the MIPS plot line coincides with a peak of the data cache misses ratio and a high value of stalls



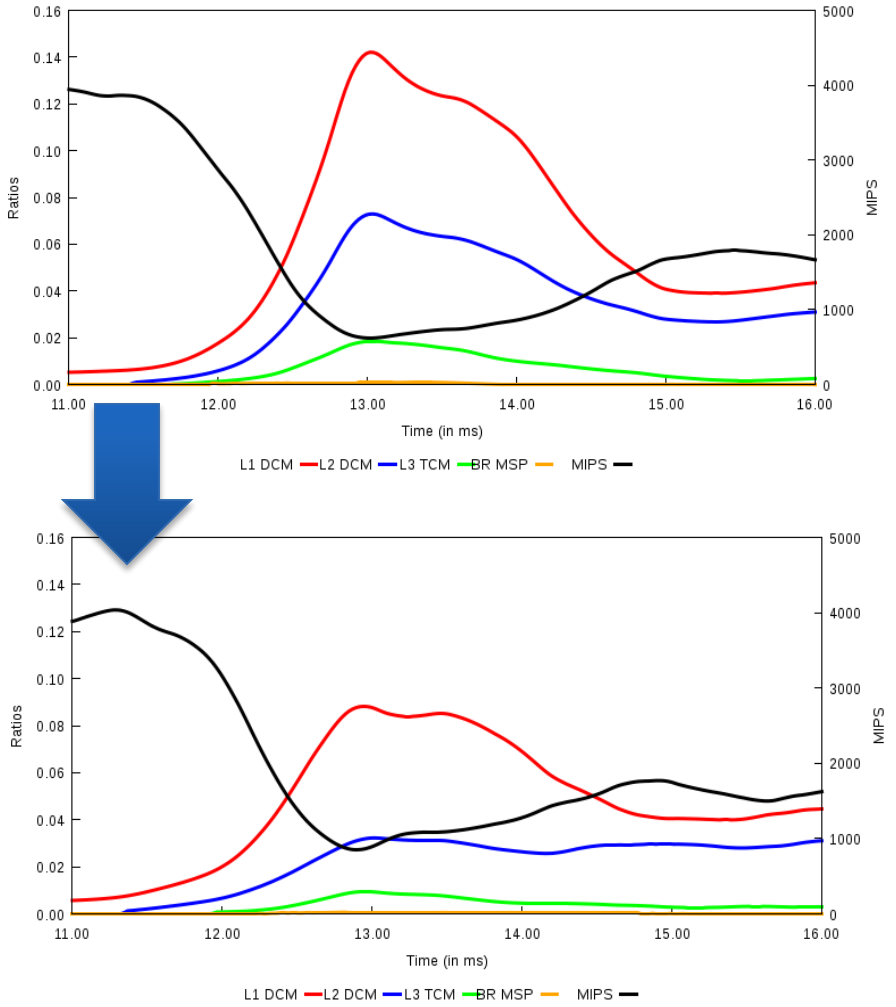


Cluster 3 – Example of better cache usage

```
DO jk = 2, jpkm1   !* Shear production at uw- and vw-points (energy conserving form)
   DO jj = 1, jpj              ! here avmu, avmv used as workspace
      DO ji = 1, jpi
         avmu(ji,jj,jk) = avmu(ji,jj,jk) * ( un(ji,jj,jk-1) - un(ji,jj,jk) )   &
            &                            * ( ub(ji,jj,jk-1) - ub(ji,jj,jk) )   &
            &           / (  e3uw_0(ji,jj,jk)         &
            &             * e3uw_0(ji,jj,jk) )
         avmv(ji,jj,jk) = avmv(ji,jj,jk) * ( vn(ji,jj,jk-1) - vn(ji,jj,jk) )   &
            &                            * ( vb(ji,jj,jk-1) - vb(ji,jj,jk) )   &
            &                            / (  e3vw_0(ji,jj,jk)              &
            &                              *  e3vw_0(ji,jj,jk)  )
      END DO
   END DO
END DO
```

```
DO jk = 2, jpkm1   !* Shear production at uw- and vw-points (energy conserving form)
   DO jj = 1, jpj              ! here avmu used as workspace
      DO ji = 1, jpi
         avmu(ji,jj,jk) = avmu(ji,jj,jk) * ( un(ji,jj,jk-1) - un(ji,jj,jk) )   &
            &                            * ( ub(ji,jj,jk-1) - ub(ji,jj,jk) )   &
            &           / (  e3uw_0(ji,jj,jk)         &
            &             * e3uw_0(ji,jj,jk) )
      END DO
   END DO
END DO

DO jk = 2, jpkm1   !* Shear production at uw- and vw-points (energy conserving form)
   DO jj = 1, jpj              ! here avmv used as workspace
      DO ji = 1, jpi
         avmv(ji,jj,jk) = avmv(ji,jj,jk) * ( vn(ji,jj,jk-1) - vn(ji,jj,jk) )   &
            &                            * ( vb(ji,jj,jk-1) - vb(ji,jj,jk) )   &
            &                            / (  e3vw_0(ji,jj,jk)              &
            &                              *  e3vw_0(ji,jj,jk)  )
      END DO
   END DO
END DO
```



Conclusions

- Instantaneous performance can provide insight information for the computation performance of a model
- The combination of the mentioned tools helps the user to find specific code parts that should be improved and which is the cause of the performance degradation.
- Specially useful with big codes with big routines where knowing that a specific routine has bad performance is not enough information.
- These tools provide the information but its user's task to get conclusions from the different metrics

Future work

- Find more areas of code with poor performance and improve them
- Contribute the NEMO developer team with code improvements identified with this set of tools
- Study the balance between sampling rate (trace files size) and results quality
- Apply these techniques to a larger range of ES codes that are used in BSC-ES

www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Thank you!

For further information please contact
miguel.castrillo@bsc.es

"Work funded by the SEV-2011-00067 grant of the Severo Ochoa Program, awarded by the Spanish Government."